Modelling Neurons

Due at 4:00pm on Monday 22 January 2018

What you need to get

- `YOU_a1.ipynb`: a Python notebook (hereafter called "the notebook")
- `hh_model.pyc`: compiled Python 3 module
- `a1_solution.pyc`: compiled Python 3 module

What you need to know

The notebook includes some helper functions that you may use. These include: `PlotSpikeRaster`, `GenSignal`, `RunHH`, and `RandomSpikeTrain`. See each function's documentation for more details on how to use it.

What to do

1. **LIF Neuron Model** [12 marks total]

   In this exercise, you will simulate the dynamic state of a leaky integrate-and-fire (LIF) neuron. The assignment code includes an incomplete version of the function

   ```
   LIF_Neuron(vin, dt, tau_m=0.05, tau_ref=0.002)
   ```

   The function has two required arguments, `vin` and `dt`, which specify the time-series of the input current, and the time step for the simulation, respectively. It also has two optional arguments, `tau_m` and `tau_ref`. They are optional because they have default values that will be used if they are not specified in the call to the function.

   The function's output has two parts, the spike times, and the sub-threshold membrane potential. See the function's documentation for more information.

   (a) [5 marks] Complete the function `LIF_Neuron` as specified. You should solve the LIF differential equation using Euler's method. Don't forget about the refractory period.

   (b) [3 marks] Run your function on a constant input for at least 1 second (of simulation time), using a time step (`dt`) of 1 ms. Create a spike raster plot. I suggest you use the supplied function `PlotSpikeRaster` (included in the notebook) for that. Label the time-axis of your plot.

   (c) [2 marks] Write some Python code that uses the function's output to determine the average spike rate in Hz (spikes per second). Express your answer in a print statement that includes both text and the numerical firing rate. For example,

   ```
   The spike rate was 19.37 Hz.
   ```

   (d) [2 marks] Run your function on random input, using a `dt` value of 1 ms. You can generate random inputs using the supplied `GenSignal` function. Plot the output of your function to produce a plot similar to Fig. 1. (The horizontal threshold line is not necessary.)

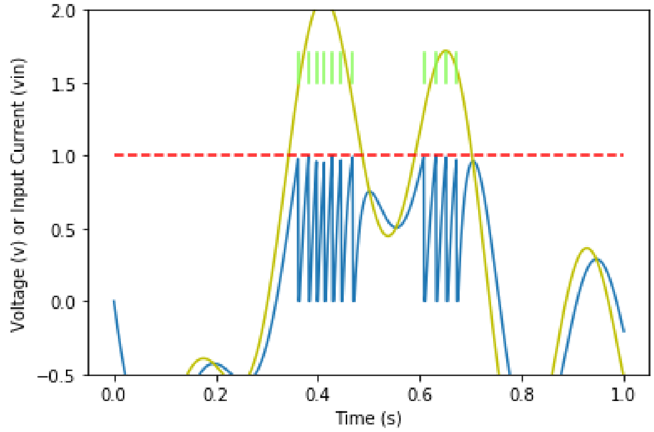

Figure 1: Example spike raster plot, also showing input (`vin`, yellow) and sub-threshold voltage (blue).

2. **Hodgkin-Huxley Firing Rate** [8 marks total]

   We saw in class that the Hodgkin-Huxley (HH) model produces action potentials, also known as "spikes". In this question, we will simulate the HH model for a series of different, constant input currents, and observe the resulting firing rates.

   For this question, you will import and use the supplied module `hh_model`, and the function `RunHH` (included in the notebook),

   ```
   t, y = RunHH(Tend, dt=0.1, Jin=0)
   ```

   See the function's documentation for more information.

   Note that the HH model uses time units of milliseconds (ms), **not** *seconds.*

   (a) [2 marks] Run the HH model for 400 ms with a constant input current in the range of 0.1 to 0.5. Produce a plot of time versus the membrane potential (v). As usual, label your axes.

   (b) [2 marks] Complete the function `CountSpikes` so that it takes the membrane potential time series (as output by `RunHH`) and counts how many times V crosses the threshold with a positive slope. See the function's documentation for an explanation of the inputs and outputs.

   (c) [4 marks] Run the HH model for a variety of input currents between 0 and 0.3 (choose about 40 different values). For each, use your `CountSpikes` function to count how many action potentials occurred, and from that the average firing rate. Plot input current versus firing rate, making sure to label the axes.

3. **Synaptic Current** [8 marks total]

   When a spike arrives at a synapse, it causes neurotransmitter to be released and dock onto receptors on the post-synaptic neuron. That docking causes ion channels to open or close, depending on the neurotransmitter and receptor type. This results in a current across the membrane of the post-synaptic neuron. The delivery of that current takes time, and is modelled by a *synaptic filter*. For a synapse with time constant τ_s, the synaptic filter of order n is

   $$s(t) = \frac{t^n \exp\left(\frac{-t}{\tau_s}\right)}{n!\, \tau_s^{n+1}} .$$

   Note that $\int s(t)dt = 1$.

(a) [4 marks] Complete the function `SpikeTrain2PSC`, which generates the post-synaptic current (PSC) for a given incoming spike train. See the function's documentation for the details of how it is to work. You may wish to use helper functions; that is fine, as long as the function `SpikeTrain2PSC` does what it is supposed to do. If you do use a helper function, you should include full documentation for how to use it.

(b) [3 marks] Using the output from the function

```
s = RandomSpikeTrain([10, 60, 20, 35], [2, 4, 6, 8])
```

plot the PSC time series for τ_s-values of 5 ms, 50 ms, and 150 ms. You can use $n = 0$, 1, or 2 for this question. Plot each on a separate plot, with samples along the t-axis every 10 ms (ie. at t-values of $\{0, 0.01, 0.02, 0.03, \ldots, 8\}$ seconds). Make sure you label the axes. You may also plot the spike raster on the same axis; you can do that using `PlotSpikeRaster`.

(c) [1 mark] What do you notice about the firing rate and the PSC as you make the synaptic time constant longer?

4. **Combining Spike Trains** [11 marks total]

Consider 3 neurons, labelled A, B, and C. Neurons A and B both synapse onto neuron C. So, the input to C is the combination of the spike trains coming from A and B. You will investigate how the synaptic time constant can affect the activity of neuron C.

The notebook uses the function `RandomSpikeTrain` to produce two identical, 1-second long, 10 Hz spike trains. The `jitter=0` argument tells the function to reduce the spike-time noise to zero.

Neuron C is a LIF neuron. You will model the activity of neuron C resulting from the spike trains arriving from A and B. If you are confident with your code for `LIF_Neuron` and `SpikeTrain2PSC`, then you can use it. Otherwise you can use the implementation supplied in the Python module `a1_solution`. Just import `a1_solution`, and call the functions using

```
a1_soluiton.LIF_Neuron( ... )
a1_solution.SpikeTrain2PSC( ... )
```

(a) [4 marks] Complete the function `SpikesAB2C`. It takes a list of two spike trains, and a synaptic time constant, and outputs the resulting spike train of the receiving LIF neuron. See the function's documentation for details.

Inside the function, you should follow these two steps:

Step 1: Compute the total post-synaptic current entering neuron C by combining the PSCs from both incoming spike trains. Denote the connection weight from A to C as w_{CA}, and the connection weight from B to C as w_{CB}. The total PSC entering neuron C would then be

$$\mathrm{PSC}_{\mathrm{total}} = w_{\mathrm{CA}}\mathrm{PSC}_{\mathrm{from\ A}} + w_{\mathrm{CB}}\mathrm{PSC}_{\mathrm{from\ B}} \, .$$

When computing these PSCs, you need to specify the sampling times. Your life will be easier if you compute the PSCs using a time step of 1 ms. You may also assume that the PSC sampling starts at a time of 0, and it is enough to go $2\,\tau_s$ beyond the last spike. For these questions, set n to 1 for all PSCs.

Step 2: Feed the total PSC from Step 1 into the `LIF_Neuron` function to get neuron C's spike train. You should use a time step of 1 ms.

Note: Let C's membrane time constant, τ_m, be 20 ms.

(b) [6 marks] Use your `SpikesAB2C` function to simulate the following scenarios.

i. **Synchronized spike trains, τ_s = 2 ms**: The spike trains from A and B are synchronized. That is, the spikes from A and B occur at the same times. Use $\tau_s = 2\,\mathrm{ms}$, and $w_{\mathrm{CA}} = 0.025$, and $w_{\mathrm{CB}} = 0.025$. Plot all three spike trains on a raster plot. Give the plot a descriptive title, and remember to label the time axis.

For the remaining questions, you can use the supplied `a1_solution.SpikesAB2C` instead of your own code.

ii. **Offset spike trains, τ_s = 2 ms**: The spike trains from A and B are offset. The spike train from A is unchanged, but the spike train from B is delayed by 50 ms. Call this delayed spike train B_{50}. Use $\tau_s = 2\,\mathrm{ms}$, and $w_{\mathrm{CA}} = 0.025$ and $w_{\mathrm{CB}} = 0.025$. Plot all three spike trains on a raster plot (do I have to mention axis labelling?).

iii. **Synchronized spike trains, τ_s = 50 ms**: Use $\tau_s = 50\,\mathrm{ms}$, and $w_{\mathrm{CA}} = 0.05$, and $w_{\mathrm{CB}} = 0.05$. Plot all three spike trains on a raster plot.

iv. **Offset spike trains, τ_s = 50 ms**: Use $\tau_s = 50\,\mathrm{ms}$, and $w_{\mathrm{CA}} = w_{\mathrm{CB}} = 0.05$. Plot all three spike trains on a raster plot.

(c) [1 mark] How does the synaptic time constant change the way C reacts to the two different sets of input spike trains (synchronized, and delayed)?

What to submit

Your assignment submission should be a single jupyter notebook file, named (`<WatIAM>_a1.ipynb`), where `<WatIAM>` is your UW WatIAM login ID (not your student number). The notebook must include solutions to **all** the questions. Submit this file to Desire2Learn. You do not need to submit any of the modules supplied for the assignment.